Engineering Note

IND360 Precision
EtherCAT PLC

EtherCAT®

METTLER TOLEDO

Menu

1. Overview

This Engineering Note is based on the integration of Mettler Toledo's Industrial Weighing Automation Terminal IND360 Precision with an EtherCAT PLC or IPC. Go to www.mt.com/ind-IND360-downloads to download all the necessary files and documents.

Note: The configuration used in this sample code is based on the default settings:

Beckhoff TwinCAT 3 PLC

SAI data format: 2-Block format

IP Address: (to be assigned by the PLC/ IPC)

IND360 device firmware version: V1.00.0016

ESI file: Mettler Toledo IND360 ESI.xml

It is recommended to start by integrating one IND360 into the PLC EtherCAT network and go through the sample code to understand the functionality of each Function Block.

2. Setup of Project Development Environment

2.1. Hardware Integration

Connect the Ethernet cable from the PLC EtherCAT port to IND360 industrial Ethernet port (X1.1 or X1.2).

2.2. Open the Sample Code

To open and use this sample project "IND360_PRECISION_ETCAT_V1_00" or solution, you need to use the BECKHOFF TwinCAT 3 – eXtended Automation Engineering (XAE) version 3.1 or higher.

Copy and paste the IND360 ESI (EtherCAT Slave Information) specification onto the TwinCAT\ 3.1\ Config\ io\ EtherCAT directory.

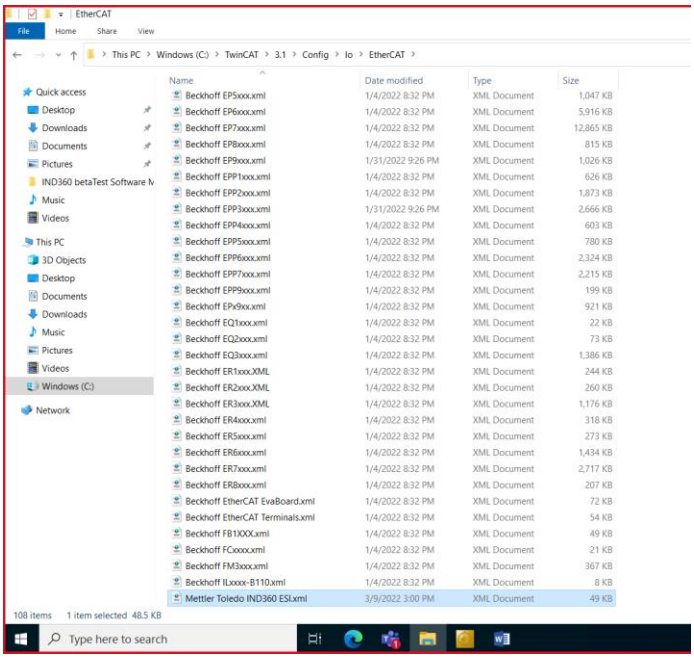

Figure 2-1: copy and paste the IND360 ESI specification

Reload the TwinCAT devices as shown below:

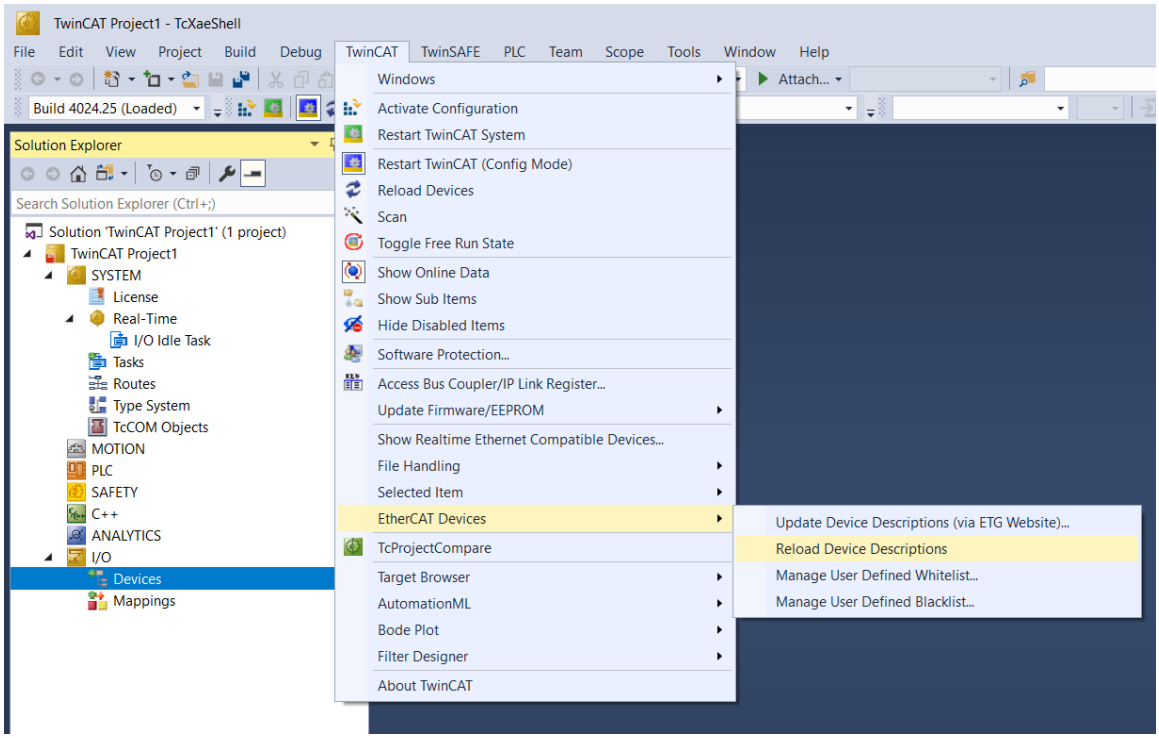

Figure 2-2: reload the EtherCAT devices' descriptions

2.3. Set up the IND360 Input/ Output Box

In this sample code, the IND360's SAI data format is the default "2 block format". This can also be configured as "8 block format" via the web browser. IND360's web configuration can be accessed by connecting the Ethernet cable to its service port with default IP address "192.168.0.8". SAI data structure will be explained in brief in Chapter 3 – SAI Data Structure. For detailed information regarding METTLER TOLEDO's SAI protocol please go to this document on MT.com.

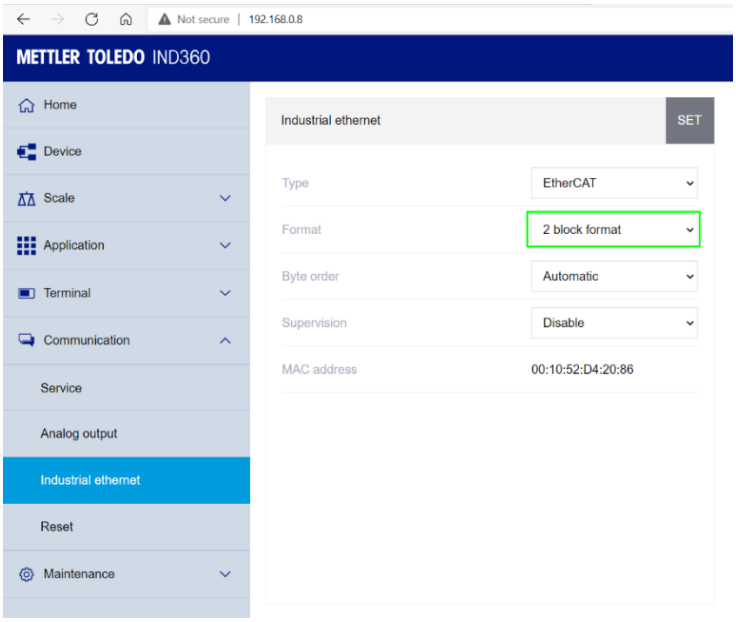

Figure 2-3: SAI data format can be 2 block or 8 block

In the TwinCAT input/ output box configuration, check the 2-Block data only. Do the same for both Inputs and Outputs.

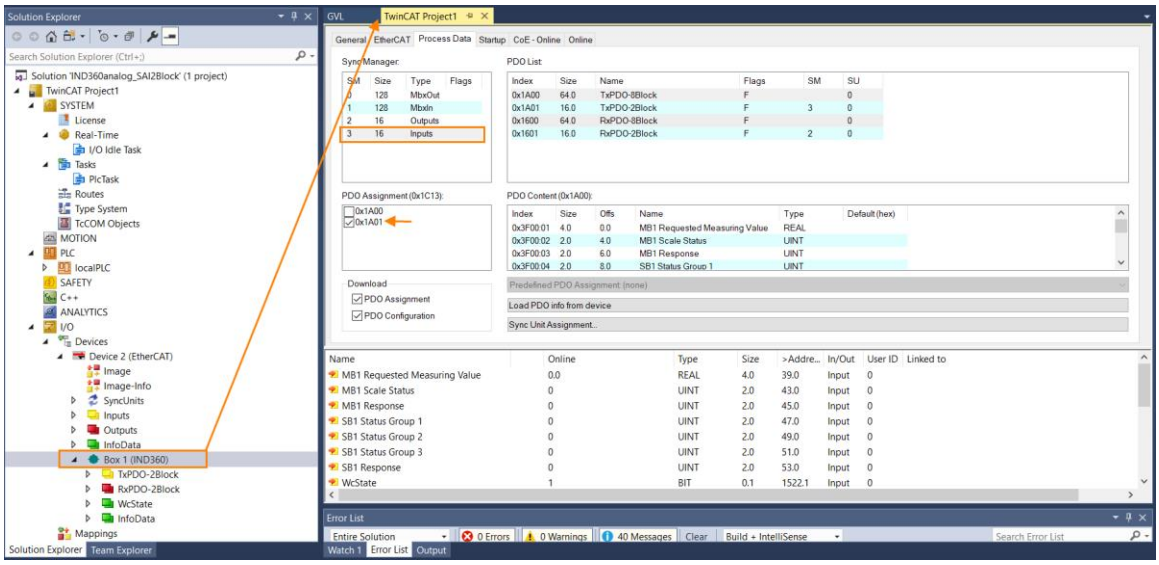

Figure 2-4: configure Inputs as 2-Block

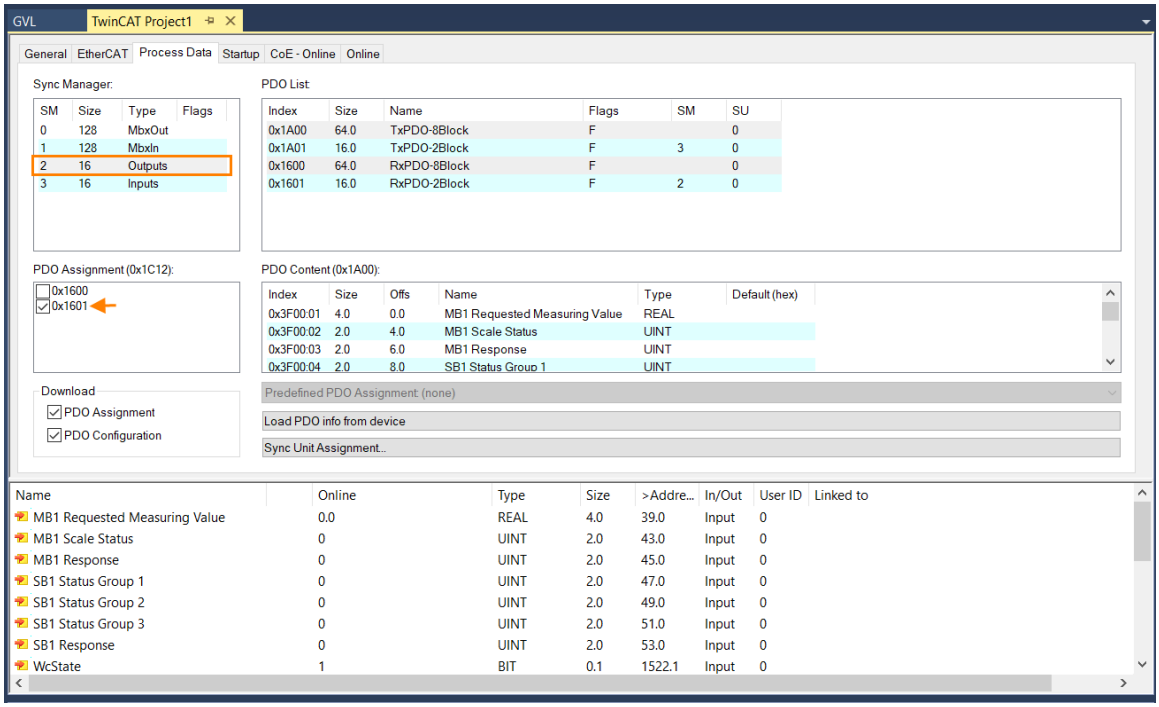

Figure 2-5: configure Outputs as 2-Block

Online reload the connected EtherCAT device and check the cyclic communication of all input/ output data.

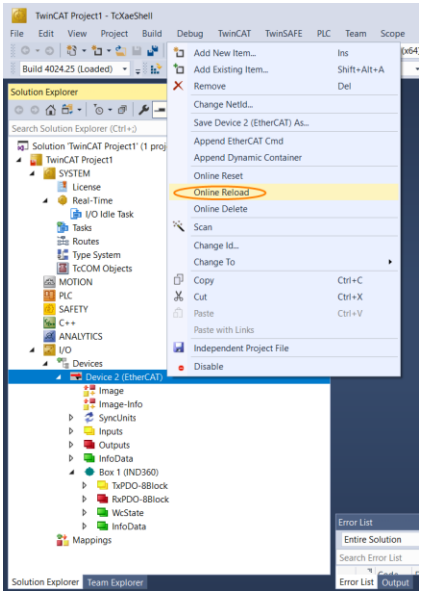

Figure 2-6: online reload the EtherCAT device

2.4. Set up the TwinCAT CoE library

Besides supporting the cyclic communication used in process control, the IND360 EtherCAT device also supports the SDO (Service Data Object) configuration via CoE interface (CAN application protocol over EtherCAT). These SDOs are mainly for scale configuration (capacity, increment, filter settings etc.) and scale calibration (adjustment).

In order to enable the SDO read/write, the TwinCAT CoE library "Tc2_EtherCAT" has to be added into the PLC project references.

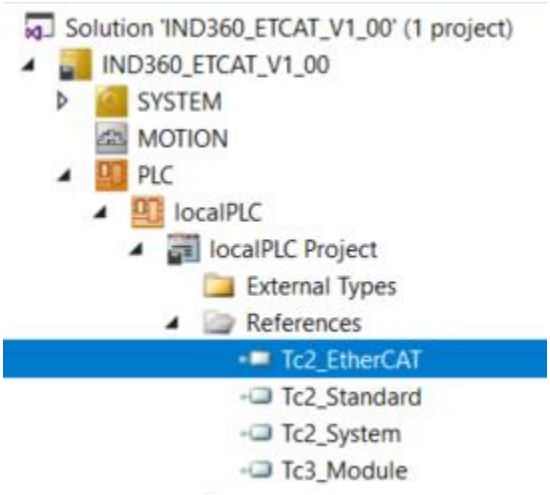

Figure 2-7: make sure the Tc2_EtherCAT is added

Function blocks "FB_EcCoESdoWrite" and "FB_EcCoESdoRead" are used in this sample code to write/ read the SDO via CoE interface. In order to use these function blocks, the EtherCAT device's NetID and Slave Address are required. These two inputs will be different for every connected EtherCAT slave.

```
fbStatus_CoESdoRead(
    sNetId      := deviceNetId,
    nSlaveAddr  := deviceSlaveAddr,
    nSubIndex   := 0,
    nIndex      := 16#4007,
    pDstBuf     := ADR(intStatus),
    cbBufLen    := SIZEOF(intStatus),
    bExecute    := bRead,
    tTimeout    := T#2S
    );
```

Figure 2-8: the EtherCAT slave's NetID and Slave Address for CoE write/read function block

The EtherCAT device's NetID and Slave Address can be found as shown below.

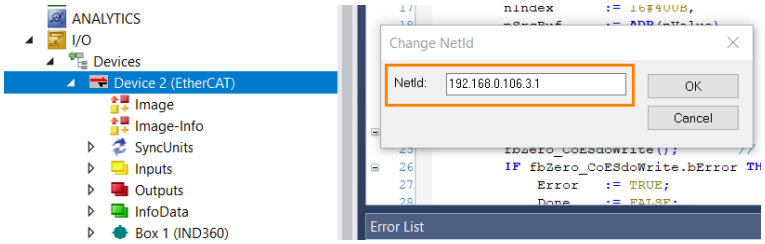

Figure 2-9: EtherCAT Device NetID

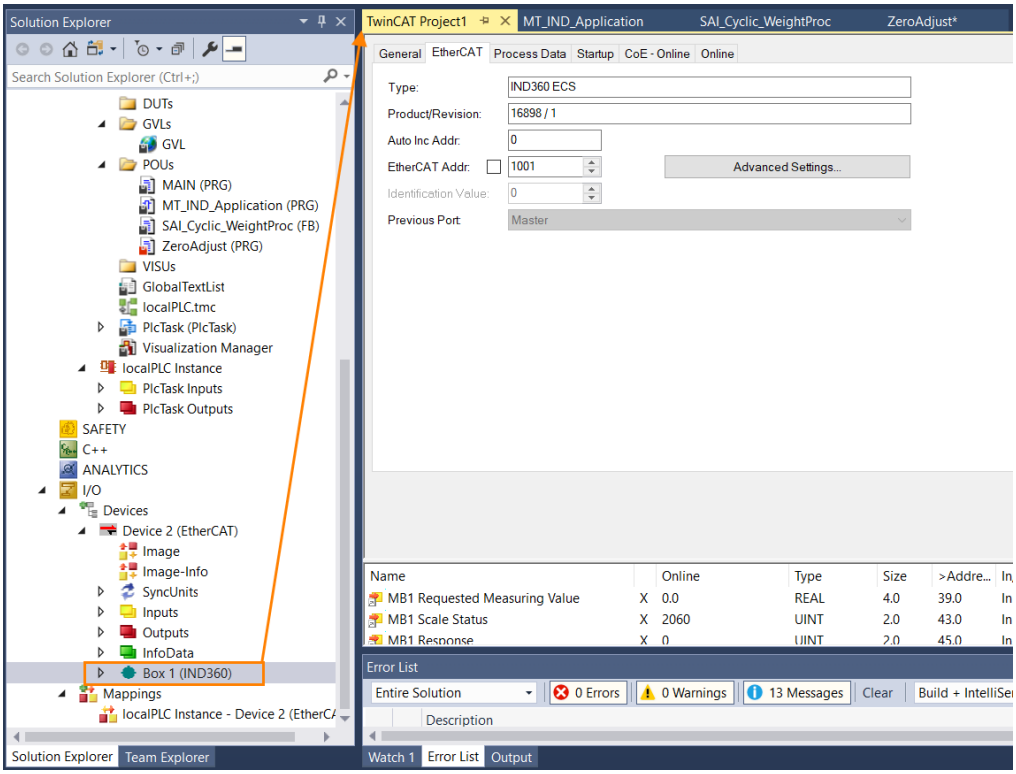

Figure 2-10: EtherCAT Device Slave Address as "1001"

2.5. Download the Sample Programming

Build the solution, then activate the configuration:

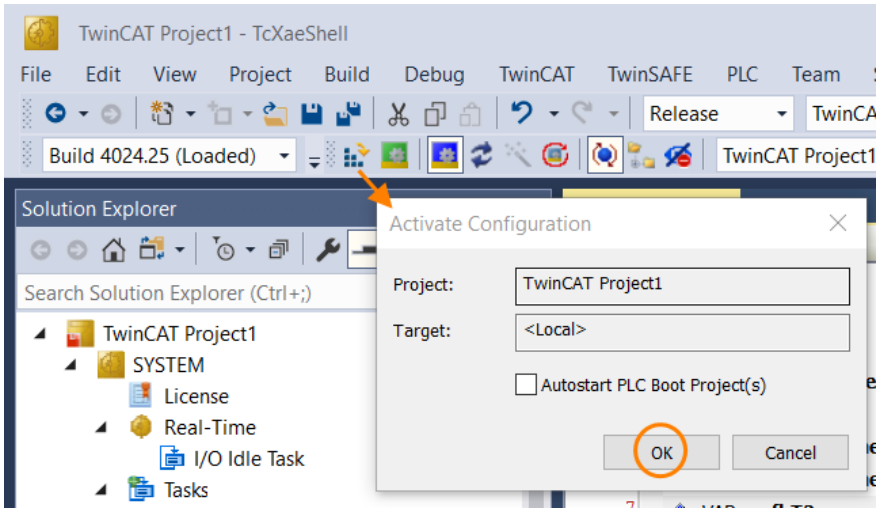

Figure 2-11: Activate the current configuration

Login to the PLC  , and run the application  .

3. SAI Data Structure

In this sample code, the IND360's SAI data format is the default "2 block format". This can also be configured as "8 block format" via the web browser. For more details on SAI data structure, please refer to the User Manual: Standard Automation Interface: IND360 Terminal English, which is downloadable from the IND360 Download Page www.mt.com/ind-IND360-downloads.

SAI 2 Block Format consists of a single Floating Point Block and a Status Block. In the figure below, the Read and Write are referring to the PLC point of view.

Floating Point Block (Read)	
Word 0	Requested floating point value (32-bit)
Word 1	
Word 2	Device status bits
Word 3	Response value

Floating Point Block (Write)	
Word 0	Floating point value (32-bit), optionally used with command
Word 1	
Word 2	Channel mask
Word 3	Command value

Status Block (Read)	
Word 0	Status Group 1
Word 1	Status Group 2
Word 2	Status Group 3
Word 3	Response value

Status Block (Write)	
Word 0	Optional Argument – word0
Word 1	Optional Argument – word1
Word 2	Optional Argument – word3
Word 3	Command value

Figure 3-1：SAI 2 Block Data Structure

The SAI 8 Block Format builds on the format structure used by the 2 Block format; providing support for eight blocks of input data and eight blocks of output data. This format was designed for applications where the users would prefer more data within one read cycle. For example, reading gross weight, tare weight and net weight all in one cycle.

The cyclic data of the 8 Block Format supports seven instances of a Floating Point Block and one instance of a Status/Command Block for each of the read and write data areas.

Floating Point Block (Read)	Floating Point Block (Write)
Status Block (Read)	Status Block (Write)
Floating Point Block (Read)	Floating Point Block (Write)
Floating Point Block (Read)	Floating Point Block (Write)
Floating Point Block (Read)	Floating Point Block (Write)
Floating Point Block (Read)	Floating Point Block (Write)
Floating Point Block (Read)	Floating Point Block (Write)
Floating Point Block (Read)	Floating Point Block (Write)

Figure 3-2：SAI 8 Block Data Structure

4. Function Blocks

Ready-to-use function blocks (FB) can be found under the POUs folder. These function blocks are being called in the program "MT_IND_Application".

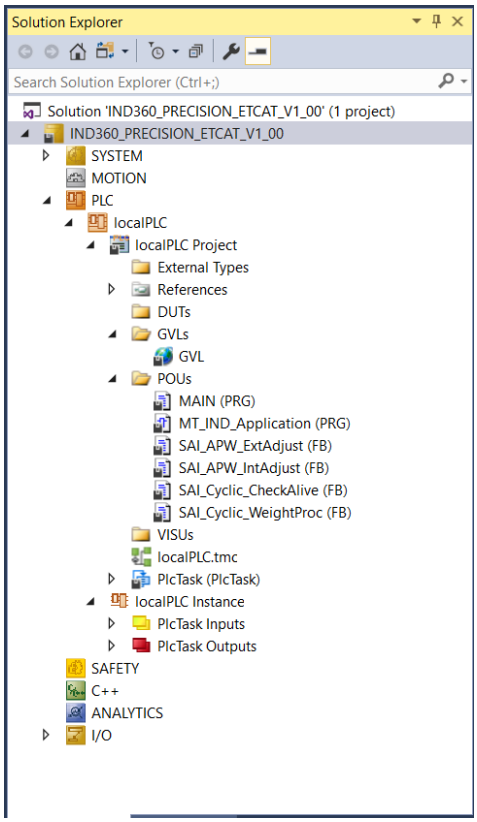

Figure 4-1: ready-to-use Function Blocks (FB) under the POUs folder

4.1. Cyclic Weight Data Processing

This function block reads in all the important real-time, cyclical weighing data such as weight value, Data OK bit, Motion bit, Net mode bit and critical alarm bit.

Set the scale command bit one at a time to trigger different commands such as tare stable, zero stable, tare immediate, zero immediate, preset tare and clear tare. A successful execution of a scale command will set the Done bit on, else the Error bit will be set on instead.

The cyclic weight data can be reported automatically right after any scale command. The type of weight data (gross, net, or tare) being reported depends on the setting for WeightCmd. By default, the WeightCmd is decimal "3" and the function block will return a net weight value every time

after any scale command such as tare or zero. Similarly, if the WeightCmd parameter is configured as decimal "0" or "1" the function block will then return a gross weight after any scale command.

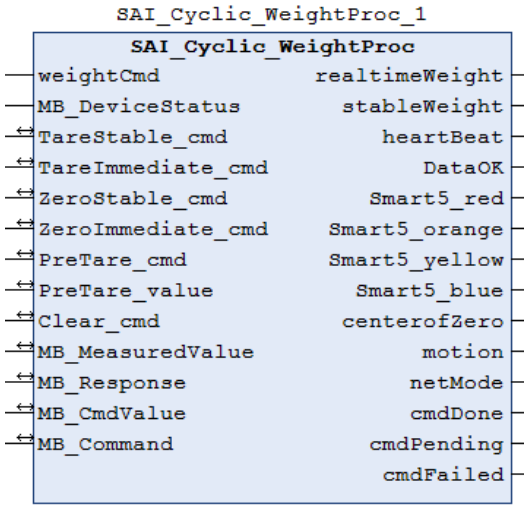

Figure 4-2: SAI_Cyclic_WeightProc Function Block

Table 4-1: SAI_Cyclic_WeightProc Function Block Parameters

Input Parameters	Data Type	Values	Description
WeightCmd	Word	0, 1	Report gross weight value
		2	Report tare weight value
		3 (default)	**Report net weight value**
		5	Report gross weight value (with internal resolution)
		6	Report tare weight value (with internal resolution)
		7	Report net weight value (with internal resolution)
MB_DeviceStatus	Word		Refer to Device Overview, input address of MB Device Status
TareImmediate_cmd	Bool		Trigger this bit to perform immediate tare command. This tare command doesn't check for stability criteria. Upon completion of this command, the input bit will be reset.
TareStable_cmd	Bool		Trigger this bit to perform stable tare command. This tare command requires the weight value to remain stable within the stability criteria (+-1d within 0.3 second) for a predefined timeout range (3 seconds by default), failing which, the command will return an error. Upon completion of this command, the input bit will be reset.
ZeroImmediate_cmd	Bool		Trigger this bit to perform immediate zero command. The zero command can only be executed when the weight value is within the zero range (+-2% by default). Else, the command will return an error. Upon completion of this command, the input bit will be reset.
ZeroStable_cmd	Bool		Trigger this bit to perform a stable zero command. This zero command requires the weight value to remain

			stable within the stability criteria (+-1d within 0.3 second) for a predefined timeout range (3 seconds by default). Furthermore the weight value has to be in the zero range to trigger this command, failing either condition; the command will return an error. Upon completion of this command, the input bit will be reset.
PreTareValue	Real		The preset tare value which has to be configured before issuing the PreTare command. Valid PreTare value is between scale's zero point up to maximum capacity.
PreTare_cmd	Bool		Trigger this bit to perform a preset tare command. The PreTareValue has to be configured prior to issuing this PreTare command. Upon completion of this command, the input bit will be reset.
ClearTare_cmd	Bool		Trigger this bit to perform a clear tare command. This command removes the tare and brings the scale into gross mode. Upon completion of this command, the input bit will be reset.
MB_MeasuredValue	Real		Refer to EtherCAT device's Box, MB1 Requested Measuring Value
MB_Response	Word		Refer to EtherCAT device's Box, MB1 Response
MB_CmdValue	Real		Refer to EtherCAT device's Box, MB1 Command Value
MB_Command	Word		Refer to EtherCAT device's Box, MB1 Command
Output Parameters	Data Type	Values	Description
realtimeWeight	Real		Real-time weight value, can be gross, tare or net weight
stableWeight	Real		Stable weight value, the last real-time weight during Motion = 0
heartBeat	Bool	0>1>0>1...	Is a security mechanism that ensures that the device is working as expected and updating data in Words 0, 1 and 2, this heartbeat bit is toggled between off and on states at a rate of 1 time per second or less. Should this bit stop toggling, this indicates that the device is no longer operating and the weight value is incorrect.
DataOK	Bool	0	This bit gets set to 0 when the device is still operational but the value being reported cannot be guaranteed to be valid. The following conditions cause the Data Okay bit to be set to 0: • Device is powering up • Device is in setup mode • Device is in test mode • Over capacity condition occurs - When the A/D converter is at its limit - Product dependent over capacity that occurs when the device determines it cannot trust the weight • Under capacity condition occurs - When the A/D converter is at its limit

			- Product dependent under capacity that occurs when the device determines it cannot trust the weight
		1	Weight data is normal, valid
Smart5_red	Bool	0/1	Also referred as Smart 5 level 5, RedAlert condition in which scale operation has to be stopped
Smart5_orange	Bool	0/1	Also referred as Smart 5 level 4, imminent failure of the scale
Smart5_yellow	Bool	0/1	Also referred as Smart 5 level 3, out of specification or wrong operator step
Smart5_blue	Bool	0/1	Also referred as Smart 5 level 2, predictive alarm or calibration/ adjustment is due
centerofZero	Bool	0/1	1 = Gross weight value is at a value of zero +/- one quarter of a weight and measures verification interval denoted as "e".
motion	Bool	0	Weight value is stable
		1	Weight value is in motion
netMode	Bool	0	Weighing is in gross mode
		1	Weighing is in net mode
cmdDone	Bool	0	Zero, tare or clear tare command is in process, or failed
		1	Zero, tare or clear tare command is successful
cmdPending	Bool	0	Zero or tare command is completed
		1	Zero or tare operation is being processed, pending due to unstable (motion) environment.
cmdFailed	Bool	0	Zero, tare or clear tare command is in process, or succeeded
		1	Zero, tare or clear tare command is not completed due to error

4.2. Device Heart Beat Monitoring

This function block monitors the Heart Beat bit of the weighing transmitter and outputs an "Alive" flag.

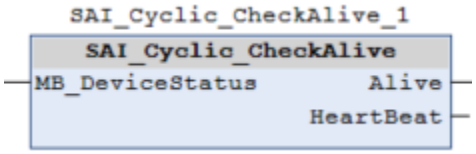

Figure 4-3: SAI_Cyclic_CheckAlive Function Block

Table 4-2: SAI_CyclicCheckAlive Function Block Parameters

Input Parameters	Data Type	Values	Description
MB_DeviceStatus	Word		Refer to EtherCAT device's Box, MB1 Scale Status
Output Parameters	Data Type	Values	Description
Alive	Bool	0	Device has lost communication
		1	Device is communicating OK

HeartBeat	Bool		To insure that the device is working as expected and updating data in Words 0, 1 and 2, this heart beat bit is toggled between off and on states. The frequency is dependent on the specific device's ability to cycle this bit. For example, a 1 second heart beat would be sufficient for most applications.

4.3. Internal Adjustment

Automated Precision Weighing "APW" module with internal adjustment feature can be adjusted without applying any external weight.

Trigger the "Start" input bit to start the internal adjustment process. Upon completion of the adjustment process, this "Start" bit will be reset.

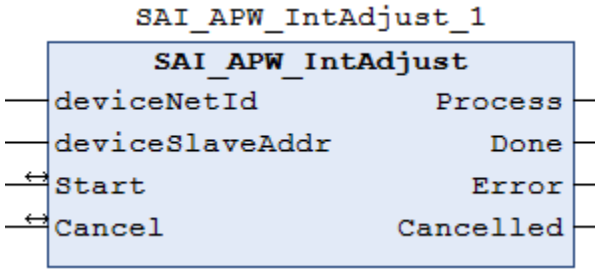

Figure 4-4: SAI_APW_IntAdjust Function Block

Table 4-3: SAI_APW_IntAdjust Function Block Parameters

Input Parameters	Data Type	Values	Description
deviceNetId	T_AmsNetId	Example: "192.168.0.106.3.1"	sNetId: String containing the AMS network ID of the EtherCAT master device. (type: T_AmsNetId)
deviceSlaveAddr	UINT	"1001"	nSlaveAddr: Fixed address of the EtherCAT slave to which the SDO upload command should be sent.
Start	Bool	1, 0	Trigger this input bit to start the adjustment process.
Cancel	Bool	1, 0	Trigger this input bit to cancel the adjustment process.
Output Parameters	**Data Type**	**Values**	**Description**
Process	Bool	1	Adjustment is started and in process
		0	Adjustment is not started
Done	Bool	1	Adjustment is completed successfully
		0	Adjustment is in process or in error state
Error	Bool	1	Adjustment failed due to error
		0	No error
Cancelled	Bool	1	Adjustment is cancelled successfully
		0	-

4.4. External Adjustment

Use this Function Block to perform scale adjustment with external weight.

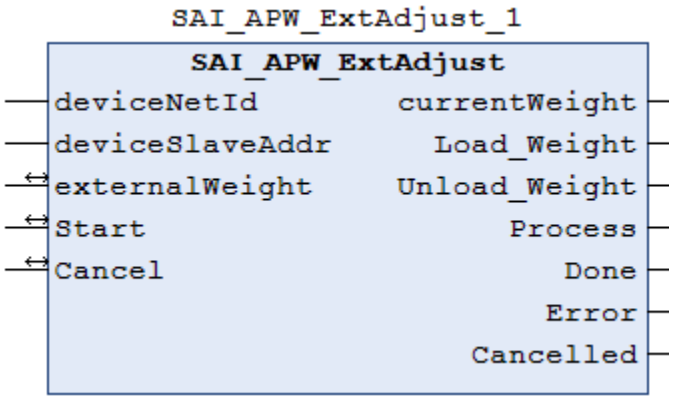

Figure 4-1: SAI_APW_ExtAdjust Function Block

Configure the "External_Weight" according to the adjustment weight to be used. The weight unit is gram by default. Trigger the Start input bit to run the adjustment procedure.

Table 4-4: SAI_APW_ExtAdjust Function Block Parameters

Input Parameters	Data Type	Values	Description
deviceNetId	T_Ams NetId	Example: "192.168.0.106.3.1"	sNetId: String containing the AMS network ID of the EtherCAT master device. (type: T_AmsNetId)
deviceSlaveAddr	UINT	"1001"	nSlaveAddr: Fixed address of the EtherCAT slave to which the SDO upload command should be sent.
Start	Bool	1, 0	Trigger this input bit to start the adjustment process.
Cancel	Bool	1, 0	Trigger this input bit to cancel/ abort the adjustment process after being started.
Output Parameters	**Data Type**	**Values**	**Description**
CurrentWeight	REAL (32 bits)	Example: "200.00"	The required reference weight here is shown here.
Load_Weight	Bool	1	User has to load the external weight according to the value displayed in CurrentWeight.
		0	No action required from the user
Unload_Weight	Bool	1	User has to unload the external weight
		0	No action required from the user
Process	Bool	1	Adjustment is started and in process
		0	Adjustment is not started
Done	Bool	1	Adjustment is completed successfully
		0	Adjustment is in process or in error state
Error	Bool	1	Adjustment failed due to error

		0	No error
Cancelled	Bool	1	Adjustment is cancelled successfully
		0	No cancellation

5. Sample Code Migration

5.1. PLC Library

As already explained in section 2.4 – Set Up the TwinCAT CoE Library, the Tc2_EtherCAT is required in order to access the SDO variables through the CoE communication.

The working PLC library is shown below.

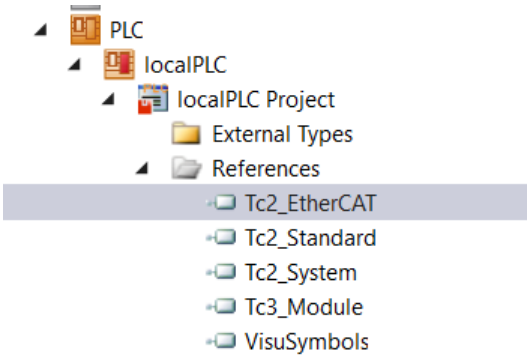

Figure 5-1: the working PLC library

5.2. Add/ Import the Global Variable List

Certain EtherCAT device input and output variables were declared in the GVL – Global Variable List. These GVLs can be copied over or imported into another TwinCAT project/ solution.

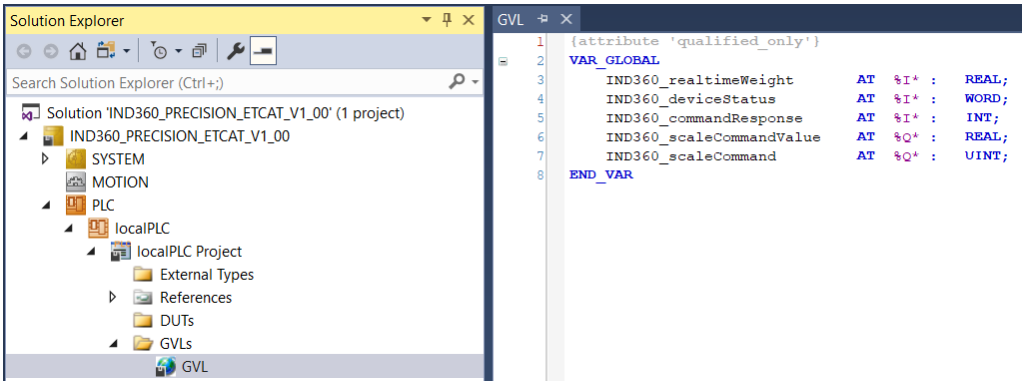

Figure 5-2: the GVLs in the sample code

These GVLs are linked to the EtherCAT device's Box as shown below.

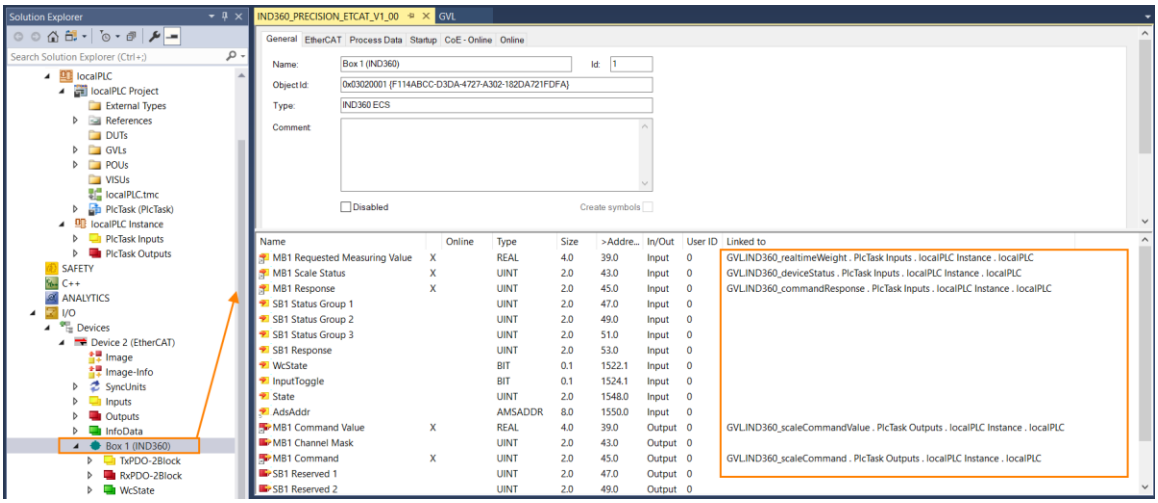

Figure 5-3: the input/ output variables being linked to the GVLs

If a new PLC variable needs to be linked to the EtherCAT device's input/ output, this can be done by right-clicking on the "Linked to" field, Change Link... and choose the correct the PLC variables.

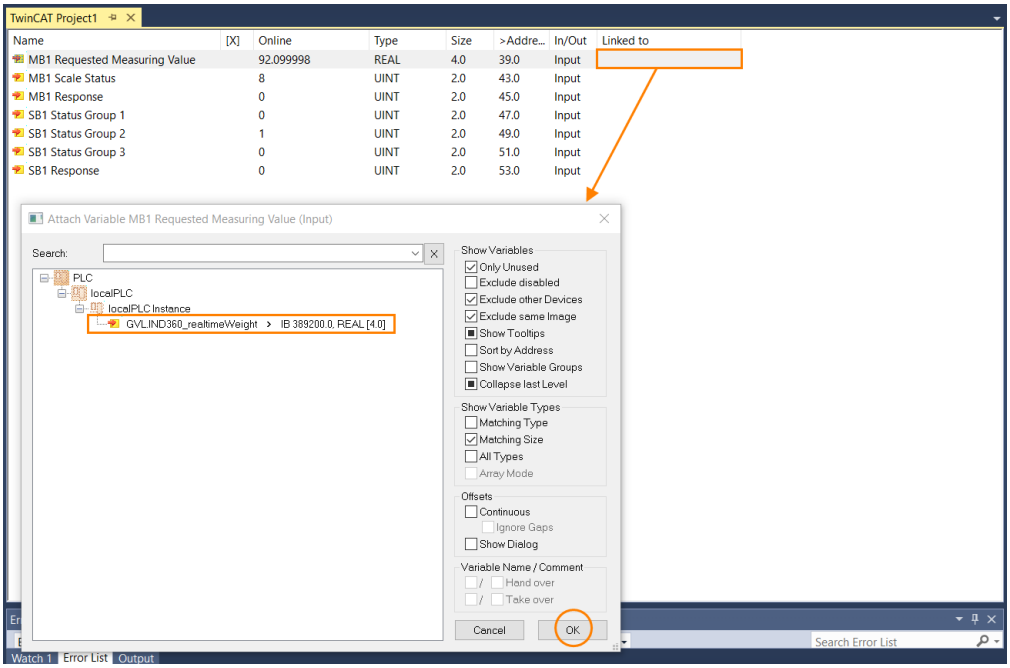

Figure 5-4: creating new link under the Box, TxPDO or RxPDO

5.3. Duplicate Programming Files

1) The required POUs:

   a) MT_IND_Application (PRG)

   b) SAI_Cyclic_WeightProc (FB),

   c) SAI_Cyclic_CheckAlive (FB),

2) The function blocks below are optional. They are used to perform scale adjustment from the PLC. Automated Precision Weighing Module (APW) can be adjusted by using the APW-Link configuration tool. Get the APW-Link software tool from www.mt.com\apw-link.

   a) SAI_APW_IntAdjust (FB)

   b) SAI_APW_ExtAdjust (FB)

   c) SAI_IND_SpanAdjust (FB)

3) MT_IND_Application is the program which runs all the Function Block instances.

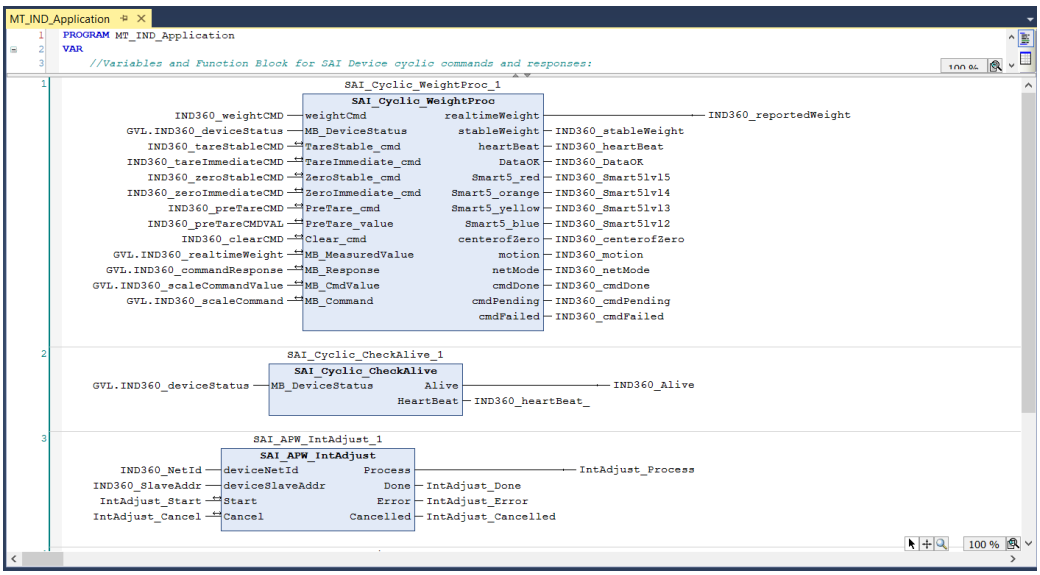

Figure 5-5: MT_IND_Application calls all the Function Blocks

4) POUs can be easily exported and imported into another solution.

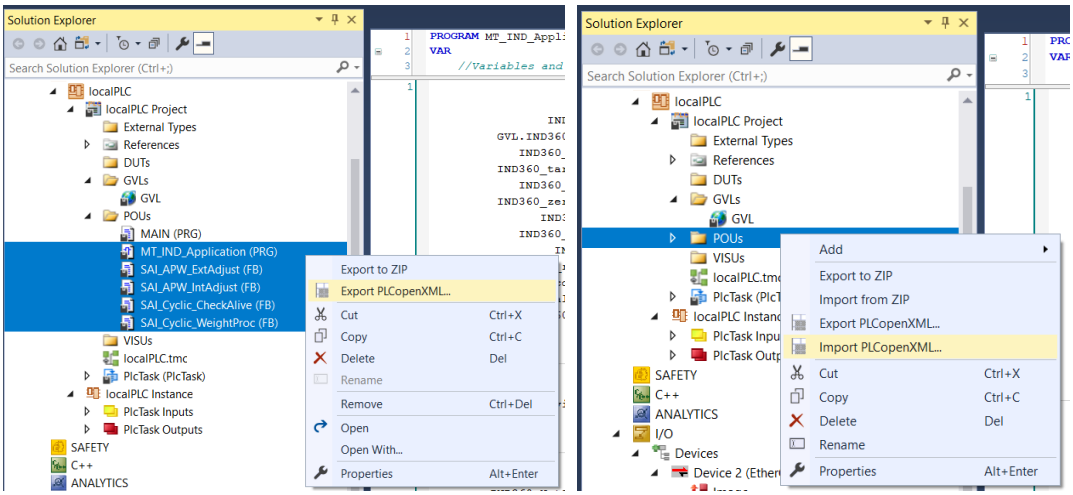

Figure 5-6: exporting and importing POUs from this sample code